UNISYS

ClearPath OS 2200

Cryptographic Library User's Guide

ClearPath OS 2200 Release 14.0

February 2013

3850 6762–001

NO WARRANTIES OF ANY NATURE ARE EXTENDED BY THIS DOCUMENT. Any product or related information described herein is only furnished pursuant and subject to the terms and conditions of a duly executed agreement to purchase or lease equipment or to license software. The only warranties made by Unisys, if any, with respect to the products described in this document are set forth in such agreement. Unisys cannot accept any financial or other responsibility that may be the result of your use of the information in this document or software material, including direct, special, or consequential damages.

You should be very careful to ensure that the use of this information and/or software material complies with the laws, rules, and regulations of the jurisdictions with respect to which it is used.

The information contained herein is subject to change without notice. Revisions may be issued to advise of such changes and/or additions.

Notice to U.S. Government End Users: This is commercial computer software or hardware documentation developed at private expense. Use, reproduction, or disclosure by the Government is subject to the terms of Unisys standard commercial license for the products, and where applicable, the restricted/limited rights provisions of the contract data rights clauses.

Unisys and ClearPath are registered trademarks of Unisys Corporation in the United States and other countries.
All other brands and products referenced in this document are acknowledged to be the trademarks or registered trademarks of their respective holders.

Contents

Section 1. Cryptographic Library

1.1. Documentation Updates 1–1
1.2. Overview of CryptoLib 1–2
1.2.1. FIPS Approved Algorithms 1–2
1.2.2. FIPS Non-approved Algorithms 1–3
1.3. FIPS Overview 1–4
1.3.1. Security Policy 1–4
1.3.2. Cryptographic Algorithm Verification Program (CAVP) 1–5
1.3.3. Cryptographic Module Verification Program (CMVP) 1–5
1.3.4. FIPS 140 References 1–5
1.4. Relationship Between CryptoLib and Communications Platform (CPComm) 1–6
1.5. New Features of CryptoLib 1R2 1–7

Section 2. How to Use CryptoLib

2.1. Installation 2–1
2.2. CRYPTOINFO UTILITY 2–2
2.3. Dynamic Linking 2–2
2.4. Using CryptoLib APIs 2–4

Section 3. Software Support Procedures

3.1. Submitting UCFs for CryptoLib 3–1
3.2. Reporting Specific Types of Problem 3–2
3.2.1. Installation Problems 3–2
3.2.2. Communications Platform SSL/TLS Problems 3–2
3.2.3. Communication Platform SNMP Problems 3–3

Appendix A. CryptoLib Security Policy

Tables

1–1. FIPS Approved Algorithms 1–2
1–2. FIPS Non-approved Algorithms 1–3

Section 1
Cryptographic Library

Cryptographic Library, also known as CryptoLib, is an OS 2200 system library containing cryptographic software that has been certified to the Federal Information Processing Standards (FIPS) 140-2 standard. The software contained in this library can be used by other OS 2200 software programs. The library is currently restricted for use by Unisys system software only.

Purpose of This Guide

This guide provides reference information for those persons responsible for administering CryptoLib.

This guide presents the following information:

- An overview of CryptoLib
- An overview of FIPS
- The relationship between CryptoLib and Communications Platform (CPComm)
- How to use CryptoLib

Audience

This guide is intended for Systems Administrators who need to install and support this product. Since this library is restricted for use by Unisys system software only, the Application Program Interface (API) for the library is not discussed.

1.1. Documentation Updates

This document contains all the information that was available at the time of publication. Changes identified after release of this document are included in problem list entry (PLE) 18893649. To obtain a copy of the PLE, contact your Unisys representative or access the current PLE from the Unisys Product Support Web site:

http://www.support.unisys.com/all/ple/18893649

Note: *If you are not logged into the Product Support site, you will be asked to do so.*

1.2. Overview of CryptoLib

CryptoLib is an OS 2200 system software library product that has been certified to the FIPS 140-2 standard. The U.S. Government and some commercial customers require FIPS 140 certification of products that use cryptography. Not all levels of CryptoLib will be FIPS 140 certified. If your site requires that FIPS 140 certified cryptography must be used, you should verify the level of CryptoLib you install is a certified level. For more information on certified levels, refer to "1.3.3 Cryptographic Module Verification Program (CMVP)." You must also verify the product is being used in accordance with the Security Policy. You can find detailed information on FIPS 140 and the Security Policy in "1.3 FIPS Overview."

CryptoLib is currently reserved for use by Unisys systems software. For this reason, the API for CryptoLib is not documented in this guide. The first user of CryptoLib is Communications Platform level 6R1. You must install CryptoLib when using Communications Platform level 6R1 or later. CryptoLib is required for generation and execution of Communications Platform.

Current FIPS 140-2 approved and tested algorithm types include symmetric encryption, asymmetric encryption, message digest, and authentication. CryptoLib includes approved and non-approved algorithms of each type.

1.2.1. FIPS Approved Algorithms

For a CryptoLib user program to run in FIPS mode, it must use only FIPS approved algorithms. The algorithms used are determined by the CryptoLib user program. Refer to the documentation of the program that uses CryptoLib to determine the algorithm selection.

Table 1–1. FIPS Approved Algorithms

Algorithm	Type	Algorithm Mode and Use
AES (128, 192, 256 bits)	Symmetric	CBC, EBC; encrypt/decrypt
3DES (168 bit effective if using 3 different keys). Note 1.	Symmetric	CBC, ECB; encrypt/decrypt
DSA (1024 bits) Note 2.	Asymmetric	Parameter generation and verification; key generation; signature generation and verification
RSA (1024, 1536, 2048, 3072, 4096 bit) Note 2.	Asymmetric	PKCS #1.5; signature generation and verification; encrypt/decrypt
SHA1 (byte only)	Digest	Message integrity
SHA2 (SHA-224, SHA-256, SHA-384, SHA-512 byte only)	Digest	Message integrity

Table 1–1. FIPS Approved Algorithms

Algorithm	Type	Algorithm Mode and Use
HMAC-SHA1	Keyed digest	Message integrity
RNG	Random Number Generator	FIPS 186-2 General Purpose [(x-Change Notice); (SHA-1)]

Notes:

1. *3DES requires the CryptoLib user to supply three keys for use by the algorithm. If all three keys are different, this is 3 key 3DES, and is referred to by the Cryptographic Algorithm Validation Program (CAVP) as KO 1 mode. If two different keys are used, it is called KO 2 mode. If all three keys are the same, it is called KO 3 mode. KO 3 mode is equivalent to DES, and as such is not a FIPS approved algorithm.*
2. *Key sizes must be equal to or greater than 1024 bits in approved mode. Select a key size that provides an appropriate balance between performance and security because the performance decreases with an increase in key size.*

1.2.2. FIPS Non-approved Algorithms

Non-approved algorithms are available to calling programs not concerned with FIPS 140 compliance. These algorithms should not be used by calling programs that wish to run in a FIPS 140 compliant mode.

Table 1–2. FIPS Non-approved Algorithms

Algorithm	Type	Algorithm Mode and Use
DES (56 bit effective)	Symmetric	CBC, ECB, CTR, CFB64 encrypt/decrypt – refer to note 1
3DES (168 bit effective if using 3 different keys) – refer to note 2	Symmetric	CTR, CFB64 encrypt/decrypt – refer to note 1
RC4 (128 bit)	Symmetric	Encrypt/decrypt
DH	Key agreement	Ephemeral
MD2 (byte only)	Digest	Message integrity
MD5 (byte only)	Digest	Message integrity
HMAC-MD5	Keyed digest	Message integrity
HMAC-SHA2	Keyed digest	Message integrity – refer to note 4
AES (128, 192, 256 bits)	Symmetric	CTR, CFB128 encrypt/decrypt – refer to note 3

Notes:

1. *CTR and CFB64 modes are first available in CryptoLib 1R2 Non-FIPS certified version.*
2. *3DES requires the CryptoLib user to supply three keys for use by the algorithm. If all three keys are different, this is 3 key 3DES, and is referred to by the Cryptographic Algorithm Validation Program (CAVP) as KO 1 mode. If two different keys are used, it is called KO 2 mode. If all three keys are the same, it is called KO 3 mode. KO 3 mode is equivalent to DES, and as such is not a FIPS approved algorithm.*
3. *CTR and CFB128 encrypt/decrypt are first available in CryptoLib 1R2 Non-FIPS certified version.*
4. *HMAC-SHA2 is first available in CryptoLib 1R2 non-FIPS certified version.*

1.3. FIPS Overview

The FIPS 140 standard is a U.S. Government security standard issued by the National Institute of Standards and Technology (NIST). It is used to accredit cryptographic modules. CryptoLib is certified to FIPS 140-2. The FIPS 140 certification program is administered by the Cryptographic Module Validation Program (CMVP) operated by NIST. The CMVP is used by cryptographic hardware and software vendors using a CMVP certified laboratory to obtain module certification. For more information on the CMVP, see "1.3.3. Cryptographic Module Verification Program (CMVP)." A certified cryptographic module must also implement at least one algorithm implementation validated by the Cryptographic Algorithm Validation Program (CAVP). For more information on CAVP, refer to "1.3.2. Cryptographic Algorithm Verification Program (CAVP)."

FIPS 140-2 defines four levels of security, "Level 1" to Level 4". Level 4 is considered the highest level of security. CryptoLib is validated at Level 1, as are most software modules. Typically, Levels 2 through 4 are for hardware cryptographic modules.

1.3.1. Security Policy

Federal Information Processing Standards Publication (FIPS PUB) 140-2 defines the cryptographic module security policy as "a precise specification of the security rules under which a cryptographic module will operate, including the rules derived from the requirements of this standard and additional rules imposed by the vendor". For software to claim it is running in FIPS certified mode, it must be a certified module, and it must be operated in accordance with the Security Policy for the module. For more information on security policies, see the links to the FIPS 140-2 Standard and the CMVP FAQ in "1.3.4 FIPS 140 References."

Refer to Appendix A for information on the CryptoLib Security Policy. The NIST Web site is the definitive source for a security policy. Security policies for all CMVP approved cryptographic modules, including CryptoLib, can be found on the NIST Web site at the following link:

http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140val-all.htm

1.3.2. Cryptographic Algorithm Verification Program (CAVP)

A cryptographic module validated to FIPS 140 must implement one or more NIST approved security functions used in an approved mode of operation. When an algorithm implementation meets all the requirements of FIPS 140, and successfully completes the CAVP, NIST issues a cryptographic module validation certificate for that algorithm implementation. The NIST Web site lists all approved algorithms, including those issued to Unisys for CryptoLib:

http://csrc.nist.gov/groups/STM/cavp/validation.html

Unisys validation certificates can be found at the above link in the validation lists for AES, Triple-DES, DSA, RSA, SHS, RNG, and HMAC.

Cryptographic algorithm validation is a prerequisite to CMVP, described in "1.3.3. Cryptographic Module Verification Program (CMVP)."

1.3.3. Cryptographic Module Verification Program (CMVP)

The CryptoLib cryptographic module has been tested and validated under the CMVP as meeting the requirements of FIPS PUB 140-2. (See the link to the FIPS 140-2 Standard in "1.3.4 FIPS 140 References.") A validation certificate has been issued to Unisys for CryptoLib. This certificate and all validation certificates can be found on the NIST Web site at the following link:

http://csrc.nist.gov/groups/STM/cmvp/validation.html

You can verify whether the version you are running is FIPS certified by checking the NIST Web site.

1.3.4. FIPS 140 References

Detailed information on FIPS 140-2 is available at the following sources.

- FIPS 140-2 Standard

  http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf

- FIPS 140-2 Derived Test Requirements

  http://csrc.nist.gov/groups/STM/cmvp/documents/fips140-2/FIPS1402DTR.pdf

- FIPS 140-2 Implementation Guidance

  http://csrc.nist.gov/groups/STM/cmvp/documents/fips140-2/FIPS1402IG.pdf

- Annex A – Approved Security Functions

  http://csrc.nist.gov/publications/fips/fips140-2/fips1402annexa.pdf

- Annex B – Approved Protection Profiles

  http://csrc.nist.gov/publications/fips/fips140-2/fips1402annexb.pdf

- Annex C – Approved Random Number Generators

  http://csrc.nist.gov/publications/fips/fips140-2/fips1402annexc.pdf

- Annex D – Approved Key Establishment Techniques

  http://csrc.nist.gov/publications/fips/fips140-2/fips1402annexd.pdf

- CAVP FAQ

  http://csrc.nist.gov/groups/STM/cavp/faqs.html

- CMVP FAQ

  http://csrc.nist.gov/groups/STM/cmvp/documents/CMVPFAQ.pdf

- FIPS 140-3 information

  http://csrc.nist.gov/groups/ST/FIPS140_3/

1.4. Relationship Between CryptoLib and Communications Platform (CPComm)

The CryptoLib product was developed to isolate cryptographic functionality of Communications Platform for obtaining FIPS 140-2 certification. The FIPS 140-2 standard requires the certification of a cryptographic module with a well-defined boundary. All cryptographic code has been removed from Communications Platform. In the future, other OS 2200 system and user software may use CryptoLib to perform cryptographic functions.

CryptoLib must be installed on the OS 2200 system. Otherwise, Communications Platform will fail to generate and it will not initialize. If CryptoLib is not installed when Communications Platform is started, a message is displayed indicating that CryptoLib must be installed, and Communications Platform will terminate. See "2.1 Installation" in Section 2, "How to Use CryptoLib," for information on installing CryptoLib. You can also find more information on how to use CryptoLib in Section 2, "How to Use CryptoLib."

1.5. New Features of CryptoLib 1R2

The following are the new features for this release of CryptoLib 1R2:

Additional CryptoLib Installation Modes

Customer Solution/Benefit:

Additional installation modes of CryptoLib are now available so a user can choose to install the latest uncertified version of CryptoLib, if desired. The FIPS certified version remains the default installation mode.

There are four installation modes.

Mode	Purpose
FIPS	Installs the current FIPS certified CryptoLib into SYSLIB*CRYPTOLIB. This is the default mode and gets installed with the fast-load tape.
FIPSTEST	Installs the current FIPS certified CryptoLib into SYSLIB*CRYPTOLIBTST. Refer to the *Communications Platform Configuration and Operations Guide* for information on using the CryptoLib Installation Utility, CRYPTO-IU, to enable your CPComm to run with SYSLIB*CRYPTOLIBTST.
NOTFIPS	Installs the latest non-FIPS certified CryptoLib into SYSLIB*CRYPTOLIB. This mode contains additional features and fixes that are not in the FIPS certified CryptoLib.
NOTFIPSTEST	Installs the latest non-FIPS certified CryptoLib into SYSLIB*CRYPTOLIBTST. This mode contains additional features and fixes that are not in the FIPS certified CryptoLib. Refer to the *Communications Platform Configuration and Operations Guide* for information on using the CryptoLib Installation Utility, CRYPTO-IU, to enable your CPComm to run with SYSLIB*CRYPTOLIBTST.

The installation modes, NOTFIPS and NOTFIPSTEST, of CryptoLib include the following features and fixes:

- Improved performance for the AES cipher and the SHA1 and MD5 digests
- Improved UC header API documentation
- Fix for 9th bits set in AES, 3DES, or DES encrypted or decrypted data
- Fix big number arithmetic library errors with 4096-bit numbers
- Added Counter (CTR) mode support to the AES, 3DES, and DES ciphers
- Added Cipher Feedback (CFB128) mode support to the AES cipher and CFB64 mode support to the 3DES and DES ciphers
- Added support for HMAC-SHA2

User Impact

If FIPS certification is not required, using the uncertified version provides the latest CryptoLib features and fixes, and ensures improved performance.

CRYPTOINFO Installation Information and Verification

Customer Solution/Benefit:

A utility called CRYPTOINFO is provided that can be used to determine what level of CryptoLib is installed, and whether the installation can be verified successfully. CRYPTOINFO is present in your CryptoLib installation file. It displays the version information of CryptoLib installed in the file, and verifies that CryptoLib can be loaded, is functional, and is authentic.

Example

```
@SYS$LIB$*CRYPTOLIB.CRYPTOINFO
CryptoLib Information Utility  2012 June 11 Mon 10:58:31
File:        SYS$LIB$*CRYPTOLIB(1).CRYPTOLIB
External ID: 1R2
Internal ID: 1-49-3
Build ID:    1-49-3
FIPS?:       No
Interface #: 5
Verification succeeded.
End CryptoLib Information Utility.
```

User Impact

The CryptoLib installation file can be verified and the installed level can be determined.

Section 2
How to Use CryptoLib

2.1. Installation

You can install CryptoLib using the Software Library Administrator (SOLAR) from a tape or Interim Correction Document (ICD) process. If any ICDs are applied to a FIPS certified version of CryptoLib, that version of CryptoLib is no longer considered certified. The source code for CryptoLib is not released because of U.S. Government export restrictions. This means no source code changes will be available to the end user. Fixes will be made available using the ICD process.

Four installation modes are provided, which allows two copies of CryptoLib to run concurrently.

CryptoLib has the following four installation modes.

Mode	Purpose
FIPS	For production – installs the current FIPS certified CryptoLib into SYSLIB*CRYPTOLIB
FIPSTEST	For testing a new level of CryptoLib – installs the current FIPS certified CryptoLib into SYSLIB*CRYPTOLIBTST
NOTFIPS	For production – installs the latest non-FIPS certified CryptoLib into SYSLIB*CRYPTOLIB
NOTFIPSTEST	For testing a new level of CryptoLib – installs the latest non-FIPS certified CryptoLib into SYSLIB*CRYPTOLIBTST

Only the FIPS mode is automatically installed in the IOE.

Refer to the *Software Products Installation Guide* for more information on CryptoLib installation modes.

Installation Tasks

To complete the installation activities for CryptoLib, perform the following steps:

1. Register the CryptoLib release tape with SOLAR. For more information, see the *SOLAR End Use Reference Manual*.
2. Install CryptoLib with SOLAR. For more information, see the *Software Products Installation Guide*.

CryptoLib is separately installable from any programs that use it, and the calling program should be dynamically linked to CryptoLib at run time. This allows calling programs to be modified in the field or new versions of calling programs to be installed, for example Communications Platform, and still use a system installed certified CryptoLib.

CryptoLib exists in a system install file that is part of the system searchable Linking System library search chain. The system defined library search chain exists in the OS2200 file SYS$*DATA$.SSDEF$. The Library Code Name (LCN) for CryptoLib is CRYPTO$LIB. This LCN and the corresponding CryptoLib system install file definition, SYSLIB*CRYPTOLIB, are added to the system library search chain when CryptoLib is SOLAR installed.

2.2. CRYPTOINFO UTILITY

The CRYPTOINFO utility can be used to determine what level of CryptoLib is installed, and whether the installation can be verified successfully. CRYPTOINFO is present in your CryptoLib installation file. It will display the information on the version of CryptoLib installed in the file, and verify that CryptoLib can be loaded, is functional, and is authentic.

Example

```
@SYS$LIB$*CRYPTOLIB.CRYPTOINFO
CryptoLib Information Utility  2012 June 11 Mon 10:58:31
File:        SYS$LIB$*CRYPTOLIB(1).CRYPTOLIB
External ID: 1R2
Internal ID: 1-49-3
Build ID:    1-49-3
FIPS?:       No
Interface #: 5
Verification succeeded.
End CryptoLib Information Utility.
```

2.3. Dynamic Linking

When a program wants to use CryptoLib, the CryptoLib initialization function CL$init must be called first. This prepares CryptoLib for use, verifies no unauthorized modifications have been made to the CryptoLib Object Module (OM), and does any FIPS mandated startup tests. On this first call to CL$init, the Linking System will load a copy of the CryptoLib OM from file SYSLIB*CRYPTOLIB into the calling program's address space for its use. The dynamic linking to CryptoLib is automatic. All calling programs get their own copy of the CryptoLib code. Because CryptoLib is a separately installable product from its calling programs, newer versions of calling programs can be installed, while leaving the existing installed version of CryptoLib unchanged.

While it is possible to statically link CryptoLib with a program, it is discouraged. This prevents the program and CryptoLib from being independently upgradeable. Also, CryptoLib initialization will likely fail unless the CryptoLib OM and OM signature are included in the program's installation file. Special steps must be taken when linking a program that uses CryptoLib to prevent it from statically linking in CryptoLib. One method is to instruct the linker to not resolve references for the CryptoLib calls. Following is an example of a link directive using this method:

```
resolve all references except
    'CL$init',
    'CL$DSA_new','CL$DSA_free','CL$DSA_size',
    'CL$DSA_load_private_key','CL$DSA_load_public_key',
    'CL$DSA_verify_signature',
    'CL$RSA_new','CL$RSA_free','CL$RSA_size',
    'CL$RSA_load_private_key', 'CL$RSA_load_public_key',
    'CL$RSA_verify_signature_ASN1',
    'CL$MD2','CL$MD5','CL$SHA1'
using LOCAL_DEFS, LCN
```

All CryptoLib API calls must be listed. The failure to list all of the calls will lead to static linking. The program can employ another method by not using 'LCN' on the 'resolve' link directive, but by specifying the files that should be used to resolve references. Following is an example of a link directive using this method:

```
resolve all references using LOCAL_DEFS,
                             SYS$LIB$*EMOMRTS,
                             SYS$LIB$*EMSRVC,
                             SYS$*RLIB$
```

The files that your program uses to resolve references can be determined by using the 'c' option on your @LINK call. You can find additional information on linking your program in the *Linking System Programming Reference Manual*.

2.4. Using CryptoLib APIs

Only Unisys system software use is supported for CryptoLib. Therefore, no API user documentation is supplied for CryptoLib. The C header files are provided in the CryptoLib install file, which contain information on each API. However, this is not an effective replacement for detailed end user API documentation. While there is no mechanism for preventing other programs from calling CryptoLib APIs, doing so would be difficult without documentation and it is strongly discouraged.

CryptoLib is not useable by calling programs until the CryptoLib initialization API, CL$init, is called. Any CryptoLib cryptographic API that is called before CL$init will receive a status indicating CryptoLib is not initialized. CL$init performs what FIPS 140 refers to as a "software integrity test." A 2048-bit RSA public key built into the CryptoLib OM is used to verify the signature of the OM. The CL$init calculated signature is compared to the signature in the CryptoLib install file to make sure the OM is not corrupted. Only the developers of CryptoLib can generate a verifiable signature. This prevents tampering with the CryptoLib OM. CL$init also performs cryptographic algorithm tests, also called Known Answer Tests (KATs), a big number library KAT, and initializes the random number generator.

Every attempt should be made in a multi-activity program to call CL$init in other than the main path of program initialization, as the function can take several seconds to complete.

If CL$init returns an error, this indicates malfunctioning hardware or corrupted software. CryptoLib will be unusable if the CL$init call returns an error. To report the problem to your designated Unisys Customer Support Center, refer to Section 3, "Software Support Procedures."

Section 3
Software Support Procedures

This section helps you decide what information to supply with an authorized User Communication Form (UCF). With the appropriate information, CryptoLib problems can be resolved more quickly.

This section provides the following information:

- Diagnostics you should include with every UCF for CryptoLib
- Special considerations when reporting specific types of problems

3.1. Submitting UCFs for CryptoLib

If you have a problem that requires Unisys customer support, contact your designated Unisys Customer Support Center. If there is no record of the problem, the support center can authorize a UCF.

Reporting a Software Problem

Problems with CryptoLib will manifest themselves as problems with the program that uses CryptoLib. It is therefore reasonable to write the UCF against the product that uses CryptoLib. Use the UCF submission recommendations for that product. If the error is clearly with CryptoLib, or the support personnel for the using product indicate, write the UCF against CryptoLib.

When submitting a UCF to report problems with CryptoLib, you should supply the following information:

- The level of CryptoLib in use, and the product using CryptoLib that encountered the error. Each CryptoLib user program gets its own copy of the CryptoLib OM dynamically linked in at run time. CryptoLib does not have its own diagnostics. It relies on the diagnostics available from the calling program.
- Diagnostic materials that are available for the product using CryptoLib (dump, trace, or log) as described in the following subsections. Diagnostic materials vary from product to product.
- A problem description
  - Identify specific external symptoms of the problem and specify by name any applications that are experiencing problems.
  - Provide a description of what happened just before the error occurred.

- Include the exact syntax of the last command entered into the calling program before the error condition.
- Provide a copy of the status or error messages printed by the calling program at the time the error occurred.
- Any keywords for the UCF should come from the list in the PVP.

Check with Support Center personnel for the information they need to best diagnose your problem and provide a solution.

3.2. Reporting Specific Types of Problem

The following subsections describe special requirements for reporting specific types of problems.

3.2.1. Installation Problems

For problems that relate to using SOLAR for installation, supply the diagnostic elements produced by SOLAR. The elements are listed at the end of a SOLAR execution.

3.2.2. Communications Platform SSL/TLS Problems

The Communication Platform implementation of the SSL/TLS protocol relies on CryptoLib for cryptographic services. Problems that relate to Communications Platform's handling of SSL/TLS may include problems with TCP or the SSL/TLS API to Communications Platform. The best course of action is to write a UCF on Communications Platform.

If the problem is in establishing a secure connection, tracing should be set to medium for TCP and SSL. If there are data corruption problems, then the component traces should be set to HIGH; however, this will have a negative impact on performance. If the problem is in the API, set SSL and API-SSL tracing to medium.

The information needed to isolate the problem should include the following:

- A process name and the type of process (for example, user application or (FTP)).
- Information about whether the process used application-imposed or Communications Platform-imposed security.
- The name of the peer software program that encountered the problem, if it is a connection problem. (Connection problems are opens or closes that do not work or sessions that hang.)
- A detailed description of the problem.
- A listing of the Communications Platform source configuration.
- Copies of the cryptographic files listed in the configuration, **except** the private key file. The private key file should always be kept secret.
- The local and remote IP addresses and port numbers if applicable.
- The vendor name and model number of the peer computer or program (for example, Microsoft Windows XP Pro, FileZilla).

- The raw Communications Platform trace and log files.
- A raw Communications Platform dump file, if possible.
- A listing of Communications Platform-applied changes.

3.2.3. Communication Platform SNMP Problems

The Communication Platform implementation of the SNMPv3 protocol relies on CryptoLib for cryptographic services. The best course of action is to write a UCF on Communications Platform. If the SNMP Management station cannot connect to the Communications Platform SNMP agent, first check that the management station and the agent have the same configured community name and it is the same case. If the community name is correct, perform a PING on the IP address of Communications Platform.

- If there is a favorable response to the PING, set SNMP and UDP traces to HIGH. Repeat the connection test, close the trace, and include this information.
- If the SNMP extension is having a problem, set the API-SNMP, API-UDP, and SNMP traces to HIGH.

Appendix A
CryptoLib Security Policy

FIPS requires every certified cryptographic module to have a Security Policy. For more information on security policies refer to "1.3.1 Security Policy." An example of the CryptoLib Security Policy is provided in this document for convenience. It might differ from the official version. The only FIPS acceptable version can be found on the following NIST Web site.

http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140val-all.htm

Unisys OS 2200 Cryptographic Library
Version 1R1
Unisys Corporation

Unisys OS 2200 Cryptographic Library FIPS 140-2
Security Policy
Version 1.08

Copyright Notice

This document may be reproduced and translated into any language freely in whole or part without the author's permission.

Unisys and the Unisys logo are registered trademarks of Unisys Corporation, in the USA.

Acknowledgements

The principal author of this document is:

James R. Heit
Consultant james.heit@Unisys.com
2470 Highcrest Road
Roseville, MN 55113

For further information contact

Mary Ann Bucher
Manager mary.bucher@Unisys.com
2470 Highcrest Road
Roseville, MN 55113

Judith Kruse
Director judith.kruse@Unisys.com
2470 Highcrest Road
Roseville, MN 55113

Unisys Cryptographic Library Module
FIPS 140-2 Level 1
Security Policy
Version 1.08

Revision Table

Revision Number	Date	Author	Description
1.0	Dec. 1, 2009	Unisys	Initial Submission
1.01	Jan. 5, 2010	Unisys	Revisions from initial CMVP Lab review.
1.02	Jan. 14, 2010	Unisys	Added table 3.1.1.
1.03	Jan. 21, 2010	Unisys	Revision from third CMVP Lab review.
1.04	Feb. 22, 2010	Unisys	Minor wording revisions.
1.05	Mar. 1, 2010	Unisys	Added CAVP Certificate Numbers.
1.06	Mar. 1, 2010	Unisys	CMVP requested corrections.
1.07	Mar. 8, 2010	Unisys	Add note signature gen/ver only approved with SHA-1.
1.08	Mar. 8, 2010	Unisys	Requested CMVP corrections.

Contents

Section 1. Introduction

1.1. Audience 1–1
1.2. References 1–1
1.3. Documents 1–1

Section 2. Specification

2.1. Overview 2–1
2.2. Specification 2–1
2.3. Boundary 2–2
2.4. Operational Mode 2–2
2.5. Validated Platform 2–3
2.6. Ports and Interfaces 2–3
2.7. Approved Cryptographic Algorithms 2–4
2.8. Non-Approved Cryptographic Algorithms 2–5

Section 3. Roles, Services and Authentication

3.1. Roles and Services 3–1
3.2. Authentication 3–3

Section 4. Physical Security

Section 5. Cryptographic Key Management

5.1. Key Generation 5–1
5.2. Key Agreement 5–1
5.3. Key Storage, Entry, and Output 5–1
5.4. Key Zeroization 5–1

Section 6. Electromagnetic Interference/ Electromagnetic Compatibility (EMI/EMC)

Section 7. Self-Tests

7.1. Power Up Tests 7–1
7.2. Conditional Self-Tests 7–1
7.3. Critical Function Tests 7–2

Section 8. Design Assurance

Section 9. Mitigation of Other Attacks

Appendix A. Glossary

Tables

2–1. Security Level per FIPS 140-2 Sections 2–1
2–2. FIPS 140-2 Approved Algorithms 2–4
2–3. FIPS 140-2 Non-Approved Algorithms 2–5

3–1. Roles and Services 3–1

7–1. Approved Cryptographic Power Up Tests 7–1
7–2. Pair-wise Consistency Tests 7–2
7–3. CRNG test FIPS 140-2 Section 4.9.2 7–2

Section 1
Introduction

This document specifies the non-proprietary Security Policy for the Unisys OS 2200 Cryptographic Library cryptographic module version 1R1. This Security Policy describes the compliance of Cryptographic Library (CryptoLib) with Federal Information Processing Standards Publication 140-2 (FIPS 140-2).

This document also specifies the required actions to use CryptoLib in a FIPS approved mode of operation. This document may be freely distributed in-whole and without modification.

1.1. Audience

This document is required as part of FIPS 140-2 validation. It describes how the Cryptographic Library Module meets the requirements of FIPS 140-2 validation.

1.2. References

This document deals only with operations and capabilities of the Module in the technical terms of a FIPS 140-2 Cryptographic Module Security Policy. More information is available on the Module from the following sources:

- The Unisys website (http://www.unisys.com) contains information on the full line of products from Unisys.
- The CMVP website (http://csrc.nist.gov/groups/STM/cmvp/) contains information on NIST and the cryptographic module validation program.

Please contact Unisys Corporation for access to proprietary product documentation for the Unisys OS 2200 Cryptographic Library

1.3. Documents

FIPS 140-2 requires a submission package containing several documents, they include:

- Security Policy, this document
- Finite State Model
- Design Documents
- Block Diagram

- Source Listings
- Vendor Evidence

Please contact Unisys Corporation for access to proprietary product documentation for the Unisys OS 2200 Cryptographic Library.

Section 2
Specification

2.1. Overview

CryptoLib is an OS2200 system software library product that has been validated to the FIPS 140-2 standard. Access to the library is provided through an Application Program Interface (API). The U.S. Government and some commercial customers require FIPS 140 validation of products that use cryptography.

2.2. Specification

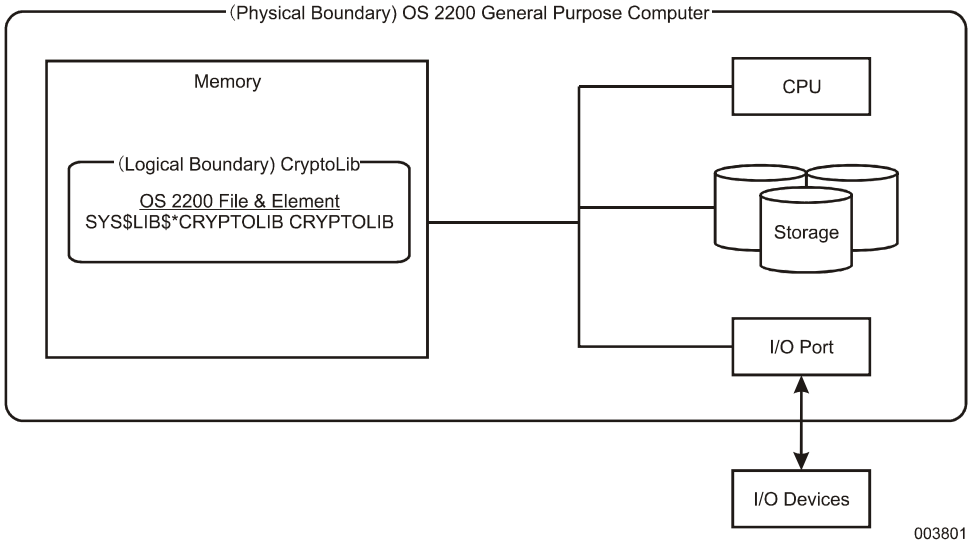
This module is classified by the FIPS 140-2 standard as a multi-chip standalone cryptographic module. This module is validated to the following FIPS 140-2 levels.

Table 2–1. Security Level per FIPS 140-2 Sections

Section	Level
Cryptographic Module Specification	1
Module Ports and Interfaces	1
Roles, Services, and Authentication	1
Finite State Model	1
Physical Security	N/A
Operational Environment	1
Cryptographic Key Management	1
EMI/EMC	1
Self-Tests	1
Design Assurance	1
Mitigation of Other Attacks	N/A

2.3. Boundary

This module is a software component only, so the logical cryptography boundary contains the software module that makes up the cryptographic module. No source code is provided to the Security Officer or User, only the binary object Module. The physical boundary includes the mainframe containing general purpose hardware including: the CPU, cache, RAM, disk drives, NICs, and other internal components of the system.



2.4. Operational Mode

When the module is uninitialized it is considered in non-FIPS mode and prevents any calls to cryptographic functions. Prior to the caller calling CL$init, any calls to cryptographic functions will return with an uninitialized status. The Module must be initialized by calling CL$init. The caller can specify whether they wish to run in FIPS Approved mode or non-FIPS Approved mode by a parameter on CL$init. The CL$init function will perform the Power Up tests, that is, the Known Answer Tests and the module integrity check. The module integrity check is done by using an RSA signature which was computed at build time. If the CL$init computed signature does not match the build signature that is distributed with the Module, the Module will not initialize and no cryptographic functions will be made available. If the Power Up tests fail, the Module will not initialize and no cryptographic functions will be made available and no data output will be output on the Data Output Interface. If the Power Up tests and Module integrity check complete successfully, cryptographic function calls become available to the User or Crypto-officer.

To summarize, to operate CryptoLib in a FIPS Approved mode of operation the following are required:

- The installed version of CryptoLib must be FIPS validated, and have no software corrections applied. Not all versions will be FIPS validated. The installation of CryptoLib is performed via SOLAR. The caller can also check a returned parameter on the CL$init call to see if the CryptoLib version in use is FIPS validated.
- The caller must specify that it wishes to run in FIPS Approved mode on the CL$init API call.
- The caller must use only approved cryptographic algorithms. See section 2.7.

2.5. Validated Platform

The FIPS 140-2 Lab tested this module on the following: UNISYS 2200 36 bit processor/ OS 2200 IOE (Integrated Operating Environment) 13.0

2.6. Ports and Interfaces

The module is a software component and utilizes Application Program Interface (API) as interfaces to the module. The module's API uses the four logical interfaces (Data Input, Data Output, Control Input, Status Output) defined by FIPS 140-2 in the following matter:

Data Input Interface

All data to all functions that is input to and processed by the Module, from the User or Crypto-officer enters via the Data Input Interface.

Data Output Interface

All functions output data (excluding statuses and return codes which are returned via the Status Output Interface) to the User or Crypto-officer via the Data Output Interface. Upon any Self-Test (Conditional or Power Up) failure the module enters an error state and the Data Output Interface is prohibited from output.

Control Input Interface

All input functions that are used to control the operation of the module enter via the Control Input Interface.

Status Output Interface

All functions provide status information back in statuses and return codes from the Module to the User or Crypto-Officer via the Status Output Interface. Some functions, such as CL$init, also provide output parameters that are defined for status output.

Power Interface

This Module is a software only cryptographic module and does not provide power or maintenance access interface beyond the power provided by the computer.

2.7. Approved Cryptographic Algorithms

The Module supports the following FIPS 140-2 algorithms in approved FIPS mode.

Note: *All Module algorithms, both approved and non-approved, are available for use. To run in FIPS approved mode, only FIPS approved algorithms should be used.*

Table 2–2. FIPS 140-2 Approved Algorithms

Algorithm	Type	Standard	Algorithm Mode and Use	Algorithm Certificate
AES (128,192,256)	Symmetric	FIPS 197	CBC,ECB encrypt/decrypt	1293
Triple-DES	Symmetric	FIPS 46-3	CBC, ECB encrypt/decrypt	910
DSA	Asymmetric	FIPS 186-2; PQG(gen) MOD(1024); PQG(ver) MOD(1024); KEYGEN(Y) MOD(1024); SIG(gen) MOD(1024); SIG(ver) MOD(1024);	Parameter generation and verification; key generation using a FIPS Approved RNG; signature generation and verification using SHA-1 only.	418
RSA	Asymmetric	PKCS#1V1.5; SIG(gen); SIG(ver)	Signature generation and verification using SHA-1 only.	619
SHA (1,224,256,384,512)	Message digest	FIPS 180-3	Message integrity	1187
HMAC-SHA-1	HMAC	FIPS 198-1	Message integrity	753
RNG	Random number generator	FIPS 186-2 General Purpose x-Change Notice w/SHA-1;	Random number Generator	721

2.8. Non-Approved Cryptographic Algorithms

The Module supports the following FIPS 140-2 algorithms in non-approved FIPS mode.

Table 2–3. FIPS 140-2 Non-Approved Algorithms

Algorithm	Type	Standard	Algorithm Mode and Use
DES (56)	Symmetric	FIPS 46-3	CBC, ECB encrypt/decrypt
MD2	Message digest	RFC1115	Message integrity
MD5	Message digest	RFC1321	Message integrity
HMAC-MD5	HMAC	RFC2104	Message integrity
RC4	Symmetric		Encrypt/decrypt
Diffie-Hellman[1]	Key agreement		
Non-approved RNG	Random number generator		Random number Generator

[1] The module only provides Diffie-Hellman primitives, which a calling application can use in a FIPS approved mode of operation as part of an allowed key establishment scheme.

Section 3
Roles, Services and Authentication

3.1. Roles and Services

There are two roles supported by Cryptographic Library, Crypto Officer and User, as defined in the FIPS 140-2 standard. The Crypto Officer and User are defined as any entity that can access the services provided by the module and each role is implicitly assumed based on the service being executed. There are no restrictions on this access. The Crypto Officer role may perform the install and uninstall of the module on the host system. The User role has access to load the module and call any API functions provided by the module.

Table 3–1. Roles and Services

Service	Role	Approved/Non-Approved
General Services		
Installation	Crypto Officer	Approved
Initialization	User	Approved
Self-Tests	User	Approved
Show status	User	Approved
Uninstall	Crypto Officer	Approved
Diffie Hellman (DH)		
Generation of key pair	User	Non-Approved
Generation of shared secret	User	Non-Approved
Digital Signature (RSA &DSA)		
DSA Key generation	User	Approved
DSA Signature	User	Approved
DSA Verification	User	Approved
RSA Key generation	User	Non-Approved
RSA Signature	User	Approved
RSA Verification	User	Approved
Digest Algorithms and Message Authentication (SHA, HMAC)		

Table 3–1. Roles and Services

Service	Role	Approved/Non-Approved
MD2	User	Non-Approved
MD5	User	Non-Approved
MD5 HMAC	User	Non-Approved
SHA 1 Digest	User	Approved
SHA 224 Digest	User	Approved
SHA 256 Digest	User	Approved
SHA 384 Digest	User	Approved
SHA 512 Digest	User	Approved
SHA 1 HMAC	User	Approved
Non-Approved Random Number Generation		
Non-Approved RNG Seeding	User	Non-Approved
Non-Approved RNG Random number request	User	Non-Approved
Random Number Generation - RNG FIPS 186-2 General Purpose x-Change Notice w/ SHA-1		
RNG Seeding	User	Approved
RNG Random number request	User	Approved
Symmetric Encryption		
AES Decryption	User	Approved
AES Encryption	User	Approved
DES Decryption	User	Non-Approved
DES Encryption	User	Non-Approved
RC4 Decryption	User	Non-Approved
RC4 Encryption	User	Non-Approved
Triple DES Decryption	User	Approved
Triple DES Encryption	User	Approved
Zeroization		
AES Zeroization	User	Approved
DES Zeroization	User	Non-Approved
Triple DES Zeroization	User	Approved
DSA Zeroization	User	Approved
MD5 HMAC Zeroization	User	Non-Approved

Table 3–1. Roles and Services

Service	Role	Approved/Non-Approved
RC4 Zeroization	User	Non-Approved
RSA Zeroization	User	Approved
SHA 1 HMAC Zeroization	User	Approved

3.2. Authentication

This Module does not support any authentication or identification services to determine the user.

Section 4
Physical Security

This module is a software library solution, and thus claims no physical security.

Section 5
Cryptographic Key Management

5.1. Key Generation

This module provides cryptographic functions for key generation. These APIs are called by applications that reside outside the cryptographic boundary. All asymmetric keys can be created for PKI, digital signing, and encryption/decryption. All keys are generated by using the approved FIPS 186-2 General Purpose [(x-Change Notice); (SHA-1)] Random Number Generator. The module does not implement a FIPS-approved RSA key generation method; however, it does make use of the approved RNG for key generation.

5.2. Key Agreement

This module provides RSA encrypt/decrypt and Diffie-Hellman primitives, which calling applications can use to implement approved/allowed key establishment methods.

5.3. Key Storage, Entry, and Output

This module does not store any critical security parameters (CSPs) in persistent state media. All CSPs generated or passed to the Module remain in the User's (calling application) memory. The User must utilize the API's correctly to guarantee FIPS 140-2 compliance.

5.4. Key Zeroization

This module does not store any CSPs and it is the User's responsibility to ensure all CSPs are deleted in a way that will make them unavailable. This Module provides functions to overwrite memory that contains keying material with zeroes. Once overwritten, the keying material will become unavailable. It is the User's responsibility to ensure the correct API is called to overwrite the keying material.

Section 6

Electromagnetic Interference/ Electromagnetic Compatibility (EMI/EMC)

This module runs on hardware that meets the applicable EMI/EMC requirements for FIPS 140-2 specified by 47 Code of Federal Regulations, Part 15, Subpart B, Unintentional Radiators, Digital Devices, Class A (i.e., for business use).

Section 7
Self-Tests

7.1. Power Up Tests

Power Up tests, also known as Known Answer Tests (KATs), are tests where a cryptographic value is calculated and compared against a result that was previously calculated and stored. KATs are not callable by Users and thus provide the User no input/output. Before any User may call into the Module they must call the initialization function CL$init which performs the KATs for approved functions. If any KAT fails, the Module will prevent any cryptographic calls from being performed.

Table 7–1. Approved Cryptographic Power Up Tests

Algorithm	Power Up Self Test
AES	Encrypt/Decrypt KAT
Triple-DES	Encrypt/Decrypt KAT
DSA	Sign/Verify Test
RSA	Sign/Verify Test
SHA	SHA-256 KAT SHA-512 KAT
HMAC	HMAC-SHA-1 KAT
RNG	FIPS 186-2 General Purpose x-Change Notice KAT
Software Integrity	RSA signature verification

7.2. Conditional Self-Tests

Conditional self-tests are executed implicitly when they are necessary. This module implements two types of conditional self tests as required by FIPS 140-2 Level 1 requirements, pair-wise consistency self-tests and continuous random number generator tests (CRNG).

Pair-wise Consistency Tests

Whenever the module creates an asymmetric public/private key pair for use by RSA or DSA, a pair-wise consistency test is performed. The module will perform a sign/verify operation on each key pair generated to ensure the key generation is functioning properly. If this operation fails, an error is reported and the key pair is discarded.

Table 7–2. Pair-wise Consistency Tests

Algorithm	Conditional Self Test
RSA	Pairwise Consistency self test
DSA	Pairwise Consistency self test

Continuous Random number generator (CRNG)

The module implements two random number generators. A non-approved random number generator (RNG), uses API calls to the OS to gather random data. The approved RNG is seeded by using the non-approved random number generator. The approved RNG is implemented based on FIPS 186-2 General Purpose x-Change Notice with the SHA-1 algorithm. This test is performed for both RNG's and upon any failure of the test an error is reported and the generated number is thrown away.

Table 7–3. CRNG test FIPS 140-2 Section 4.9.2

Algorithm	Conditional Self Test
Non-approved RNG	CRNG test FIPS 140-2 Section 4.9.2
RNG	CRNG test FIPS 140-2 Section 4.9.2

The following additional Conditional Self-Tests are not applicable to this module:

Bypass Conditional Self-Test (Not Applicable)

This module does not support a bypass capability.

Firmware Load Conditional Self-Test (Not Applicable)

This module does not reference any externally cryptographic modules or devices.

Manual Key Entry Conditional Self-Test (Not Applicable)

This module does not allow keys to be manually entered.

7.3. Critical Function Tests

This module does not implement any critical function tests for FIPS 140-2 Level 1.

Section 8
Design Assurance

Unisys manages and maintains source code and associated User documentation using the PRIMUS source control system. PRIMUS is also used for product build management, and tracking which versions of the files are used in each release.

Section 9
Mitigation of Other Attacks

This module has no prevention against specific attacks made on the module.

Appendix A
Glossary

Abbreviation	Expanded Form
AES	Advanced Encryption Standard
API	Application Programming Interface
CBC	Cipher Block Chaining
CMVP	Cryptographic Module Validation Program
CryptoLib	Unisys OS 2200 Cryptographic Library
CSP	Critical Security Parameter
DES	Digital Encryption Standard
DH	Diffie-Hellman
DSA	Digital Signature Algorithm
ECB	Electronic Code Book
EMC	Electro Magnetic Compatibility
EMI	Electro Magnetic Interference
FCC	Federal Communications Commission
FIPS	Federal Information Processing Standard
HMAC	Hashed MAC
KAT	Known Answer Test
MAC	Message Authentication Code
MD2	Message Digest Algorithm 2
MD5	Message Digest Algorithm 5
NIST	National Institute of Standards and Technology
RSA	Rivest-Shamir-Adleman
RNG	Random Number Generator
SHA	Secure Hash Algorithm

© 2013 Unisys Corporation.
All rights reserved.

3850 6762–001